Basic Artificial Intelligence Techniques

Machine Learning and Deep Learning

Bradley J. Erickson, MD, PhD

KEYWORDS

• Deep learning • Convolutional neural network • U-net • Feature engineering

KEY POINTS

- There has been and will continue to be rapid advances in deep learning technology that will have a positive impact on medical imaging.
- It is critical to understand the unique properties of medical images and the performance metrics when building a training set and training a network.
- Medical images have some unique properties compared to photographic images which require adaptation of models built for photographic imaging.

MEDICAL IMAGES

An important starting point in the application of machine learning and deep learning, particularly for those unfamiliar with medical imaging, is to recognize some of the special properties of medical images. Compared with photographic images, medical images typically have lower spatial resolution but higher contrast resolution. For instance, MR images typically are 256 × 256 and computed tomography (CT) images are 512 ×512. CT images have at least 12 bits of grayscale information, which is more than what the eye can perceive; therefore, perceiving all the information present in a CT image requires that multiple contrast and brightness settings be used. In the radiology world, these are referred to as the window width and window center. Essentially, the intensity ranges from center−width/2 up to the center + width/2. CT images have a constant intensity meaning, such that water has a value of 0 and air is −1000. And the units are Hounsfield units, honoring the inventor of the CT scanner, Sir Godfrey Hounsfield. On the other hand, other medical imaging types typically do not have a reproducible intensity scale; therefore, intensity normalization is a critical early step in the image processing pipeline.

It was recognized in the late 1980s that an open standard for representing and transmitting medical images was critical to the advance of the medical imaging sciences. That led to the development of the American College of Radiology (ACR) National Electrical Manufacturer Association (NEMA) standard, which subsequently became known as Digital Imaging and Communications in Medicine (DICOM) rather than ACR-NEMA, version 3. DICOM images consist of a header and a body, where the actual pixels of the image are the body. The header consists of several keys and values, where the keys are a set of standard and coded tags and the values are encoded in a prescribed way. DICOM tags typically are referred to as having a group and an element, each consisting of 4 hexadecimal digits. The NEMA Web site[1] contains the entire dictionary of legal DICOM tags. Thaicom does permit manufacturers to insert proprietary information in any tag where the group number is an odd number. This allows storage of information of interest to the image generator and/or not yet standardized by DICOM.

Department of Radiology, Mayo Clinic, Mayo Building East 2, 200 First Street Southwest, Rochester, MN 55905, USA
E-mail address: bje@mayo.edu

Radiol Clin N Am 59 (2021) 933–940
https://doi.org/10.1016/j.rcl.2021.06.004

HOW IMAGES ARE STORED

The original focus of DICOM was focused on the transfer of image data between 2 devices; therefore, a file format was not included in the original specification. Even today, much of the DICOM standard focuses on transfer of the image data rather than storage. There is a DICOM standard, however, for how image data should be stored, which essentially is a serialization of the header and body. Most of the time, each 2-dimensional image is stored as a separate DICOM file although standards do exist for both multidimensional and multi–time point images. Adoption of these letter formats has been slow.

Particularly before these multidimensional formats existed, medical imaging researchers developed their own formats for storing images. One of the early popular formats is referred to as the analyze[2,3] format. it had 1 file for the header information, which described the image data, whereas the other file was the actual pixel data. The Neuroimaging Informatics Technology Initiative (NIfTI) format extends the analyze format to provide more information in the header and also to join the 2 components into 1 file. There are other formats, such as mhd and nrrd, which are similar to NIfTI and supported by some specific software packages.

ELEMENTS OF MACHINE LEARNING

Features

A machine learning system requires features that are numerical values computed from the image(s). When several such values for 1 example are put together, they are called a feature vector. For a system to learn, it must be given the answer for each of these examples, and it must be given a reasonable number of examples. The number required depends on how strong the signal is in the features as well as the machine learning method used.

Features are the real starting point for machine learning. In cases of medical images, features may be the actual pixel values, edge strengths, variation in pixel values in a region, or other values computable from the pixels. Nonimage features, such as the age of the patient and whether a laboratory test has positive or negative results, also can be used. When all these features are combined for an example, this is referred to as a feature vector or input vector.

Feature engineering

Although these features might make it sound like the native pixel values simply could be used as the features, this actually is rare. The intensity often is 1 part of the vector, but other features, such as edge strength, regional intensity, regional texture, and many others, routinely are used. Determining what should be used and how to calculate those from medical images is feature engineering. Good feature engineering requires knowledge of the medical image properties (eg, how to normalize intensities and whether pixel dimensions are fixed, calculable, or unknown) and knowledge of image processing algorithms that can extract the features likely to be useful.

Feature reduction

In general, machine learning benefits from having more data for each example (as well as having more examples) in order to learn the task. It also is the case, however, that including features that either do not help in making the prediction (the feature is not informative) or that overlap with other features can result in poorer performance. Therefore, it generally is desirable to remove noncontributory features and also those that do not contribute significantly, a process known as feature reduction and as feature selection. Feature reduction has the additional benefit of reducing the computational cost at inference time.

There are 3 categories of methods used to achieve feature reduction: filter methods, wrapper methods, and embedding methods. Filter-based methods use some metric to determine how independently predictive a given feature is, and those features that are most predictive while also being independent of others are selected. Pearson correlation and chi-square are 2 popular filter methods.

Wrapper methods search for those features that result in the least reduction in performance when some feature is removed. As a wrapper method proceeds, it continually tries removing features, removing those that either are not predictive or that overlap significantly with other features.

Some learning methods have feature reduction built into them, and thus the term, *embedded*. Examples of embedded methods include lasso and random forests, where the process of training includes removal of features that do not significantly improve performance.

Machine Learning Models

Before discussing the actual machine learning techniques, the use of the term model should be addressed. A model can refer to the general shape of the machine learning method, such as a decision tree or support vector machine (SVM), and also to the specific form of a deep learning network, such as ResNet50 or DenseNet121. Model also can refer to the trained version of the machine learning tool, so readers must infer from

context which meaning of model is meant by an investigator. Because this article does not describe any trained versions, model always refers to the architecture and not to a trained version.

Logistic regression

Logistic regression is a well-established technique which, despite its name, is used more generally as a classifier. Logistic regression models have a fixed number of parameters that depend on the number of input features, and they output categorical prediction. It is similar to linear regression, where several points are fitted to a line, minimizing a function like the mean squared error (MSE). Logistic regression instead fits the data to a sigmoid function from 0 to 1, and, when the output is less than 0.5, the example is assigned to a class, else it is the other.

Decision trees and random forests

Decision trees get their name because they make a series of binary decisions until they make a final decision. In the simplest case, a range of values is tried and the best threshold is the one that gets the most cases right. Just 1 such branch usually is too simple for practical uses. The training process consists of determining which feature to make a decision on (eg, age of the subject) and what the criterion is (eg, Is age >68?). The metric used for selecting the feature usually is the Gini index or the entropy (also called information gain) for categorical decision trees or the mean error or MSE for regression trees. These metrics all focus on finding the feature that improves the most in making a prediction. Once that feature is determined, the threshold/decision criteria are computed. This process of finding the feature and criterion is applied recursively to each of the groups that result from applying the split until some stopping criterion is met (eg, no more than X decisions or until there are fewer than Y examples after a decision is applied).

An important advantage of decision trees is how easy they are to interpret. Although other machine learning models are close to black boxes, decision trees provide a graphical and intuitive way to understand what the ML model does.

Support vector machines

SVMs are based on the idea of finding a hyperplane that best divides the set of training examples into 2 classes. Support vectors are the examples nearest to the hyperplane, the points of a data set that, if removed, would alter the position of the dividing hyperplane. A hyperplane is a line that linearly separates and classifies a set of data. The goal then is to determine the formula for a plane that best separates the examples. This is called a hyperplane because the dimensionality of the plane is the dimension of the examples (and remember each example is a vector of features). It is common to remap the points from simple n-dimensional space to a different type of space if that can produce a better separation of points. There also are hyperparameters (a variable that is external to the model and whose value cannot be estimated from data) that have an impact on how a model develops. For instance, in cases of SVMs, a penalty must be assigned to an example that is on the wrong side of the decision plane. The hyperparameter is the weighting of that penalty—the weighting of no examples really wrong (therefore, assigning a high power to the error) versus fewer examples wrong, even if those are really wrong.

Neural networks

Neural networks get their name because they are modeled on how it is believed the neurons of the brain work. A neural network has an input layer, an output layer, and a variable number of middle (hidden) layers. The input layer of course receives the example vector—each entry in the vector goes to one neuron (hereafter referred to as a node). Each node then applies an activation function, similar to how a biological neuron fires given a strong enough input. The output of each node then is passed to every node of the next layer, but there is a weight for each of these connections that alters the strength of that signal. Each node in this layer applies its activation function and in turn passes their outputs to the next layer after applying a weight. This continues until the output layer. Although earlier versions used sigmoidal functions because they were similar to biological neurons, it is common to use simpler functions like rectified linear functions (eg, values below a threshold become 0, and values above the threshold are passed through).

Although the activation function is predetermined, the weights are actively learned. A common way those weights are learned (that is, altered until the network makes good predictions) is by applying back propagation. Back propagation repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector.[4] When the prediction is very wrong and getting worse compared with prior example, the weights are altered more and in the opposite direction, and, when the prediction is getting better, the weights are altered less and in the same direction as before. An important challenge is deciding which weights should be adjusted, because adjust all

weights probably should not be adjusted the same amount.

DEEP LEARNING

As discussed previously, neural networks are a machine learning technique that has been around for many years, but they were never successful. Attempts to have more than 1 hidden layer resulted in not just significant computing demands, but algorithmic challenges in how to update the weights. As a result, they fell into disfavor with SVMs and decision tree methods becoming much more popular.

Some scientists did continue to work on them, however, and made a splash in 2012 when they crushed the competition with their deep network system.[5,6] There appear to be a few factors that came together that contributed to this accomplishment. Although computers were benefiting from Moore's law, deep learning in particular benefitted from speed improvements much greater than Moore's law because the deep learning computations were mapped onto graphical processing units, which had hundreds to thousands of cores, versus the 4 to 8 cores present in a typical central processing unit. Perhaps more important were theoretic advances that made these calculations work. These included better back propagation methods, which was a critical element of updating the weights in a multilayer neural network. It also was the combination of convolutional layers with the traditional neural network as well as other specialized layers that also made a difference.

Convolutional Neural Networks

These first winning deep learning networks were convolutional networks, and they still enjoy much success today, albeit with some modifications. Although there are many variants of convolutional neural networks (CNN)s , such as AlexNet, VGGNet, GoogLeNet, and so on, all of these start with the first layers consisting of convolutions that are passed over the image. The size of the kernel varies, although more modern architectures usually use 3 $\times$ 3 kernels. The next layer after the convolution is a pooling layer, where the output of the kernel typically is reduced in size (eg, a 2 $\times$ 2 is reduced to a single pixel), most often by using the maximum value of the output is taken (MaxPool). This reduced resolution image then has another kernel applied, again followed by a MaxPool. These layers typically are finding low-level features (eg, lines and edges) in early layers and as pooling reduces resolution and combines these low-level features together to find higher-level features (eg, complete circles, eyes, or noses). After a few of these convolution and pooling layers, the output typically is flattened to a 1-dimensional vector, and that vector then is passed into a fully connected network—the familiar neural network structure (**Fig. 1**).

Fig. 2 provides a visualization of the actual content passing from layer to layer. Each output node usually maps to a class, so that if trying to predict an image as 1 of 5 classes, there are 5 output nodes. The output of the fully connected network then often is passed through a softmax function, which adjusts the output values so that they sum to 1.0, thus making the output of each node in the output layer akin to a probability.

A critical element of making such a network useful is that the weights must be adjusted so that the predictions are accurate. The most common way of doing this is by a process called back propagation. Before a network is trained, and an example is passed into the CNN (or any neural network), the output most likely is wrong. Wrong is defined as the sum of the outputs of the nodes: for the node that was the correct class, the output should have been 1 and all others 0. The error thus is the sum of 1 minus the output of the correct node plus the outputs of all the other nodes. This error then is used to guide the adjustment of the weights in the network.

Another critical element of success in training a network is normalization of the input, It can be imagined that if some images are very bright, and others are very dark, that this could confuse a machine learning tool, because it might focus on global intensity as signal rather than the other aspects that are more important. Although a few medical modalities like CT have a reproducible intensity scale, most do not. That means that normal brain tissue could have an intensity of 100 for 1 scan and 1000 on another. A common approach to normalization is to set the mean value to 0 and the range set such that 1 would has a magnitude of 1. This can work for CT as well, although it might mean that important information is lost.

Normalization also may be applied within a network, sometimes at the layer level and sometimes at the batch level. This can help avoid cases where values spiral to very high or very low levels that can exceed the numerical accuracy of the processors used, and thus produce less predictable training.[7]

Another important advance in network training that is less intuitive is dropout.[7,8] Dropout is the random removal of nodes from a network. This can substantially reduce the chance of overfitting,[7–9] apparently because the removal of nodes

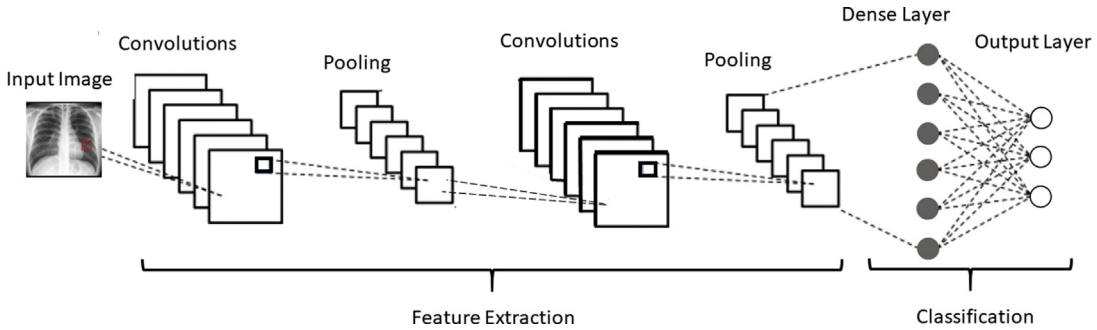

Fig. 1. A simple CNN applies convolutions to small regions of an input image, typically pooling the output of the convolution to select the important features and reduce resolution, until the last pooling layer where the output is flattened into a 1-dimensional vector that is used as input to a dense layer. There may be more that 1 of these densely connected layers, but, ultimately, there is an output layer for the prediction of the CNN.

forces the remaining nodes to maximize the contribution to the prediction.

Another technique that can reduce the chance of overfitting is the application of residual blocks (also called residual networks).[10,11] Residual blocks essentially force each layer of a network to learn—to contribute to reducing error—by having a skip connection around that layer that is the identity function. If the skipped layer(s) do worse than identity, they are ignored, and the identity function output is used. This emphasizes the point that layers cannot keep being added and performance improved out to infinity—there is a point where adding layers becomes counterproductive, but residual blocks help reduce sensitivity to this situation.

Although most of this discussion is focused more on the problem of classification, deep learning also can be useful for performing image segmentation—the assignment of labels to pixels within an image to indicate what they are (eg, pixels of the liver or brain or lung tumor). The

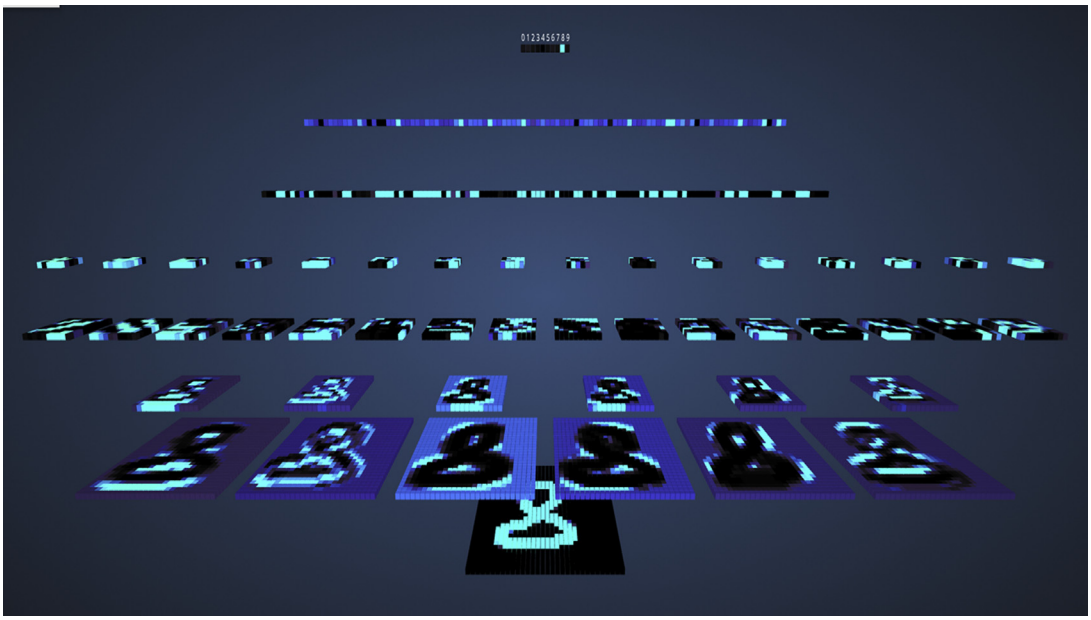

Fig. 2. Visualization of the features used for identifying a hand-drawn numeral. Readers are encouraged to access the Web site used to generate this figure and interactively see the features recognized and ultimately flattened to a 1-dimensional vector used to predict the numeral. The Web site is https://www.cs.ryerson.ca/~aharley/vis/conv/.

U-net is an important technique for image segmentation, because it performs better than most other machine learning techniques. It first was described in 2015 by Ronneberger and colleagues[12] and gets its name because the network diagram looks like a U. The basic concept is that in 1 arm of the U, the resolution of the image is decreased whereas the information about the layer (also known as filters) usually increases to some lowest resolution level. Then resolution is restored when passing up the other arm of the U, but there are skip connections that help produce a high-quality segmentation. Since the original description, many variations of U-Net have been described, including a volumetric (3-dimensional) version called a V-Net.[2]

Assessing Performance

There are many ways to measure the performance of an artificial intelligence (AI) tool. A simple one is accuracy—which percentage of cases are predicted accurately? But accuracy is sometimes misleading. For instance, if trying to determine whether there are any images of a giraffe in some collection of images, and if the number of giraffe images is quite small (say <1%), an algorithm that says there are no giraffe images would be greater than 99% accurate. For this reason, the author usually reports multiple measure, including sensitivity (true positives/all positives) or specificity (true negatives/all negatives) or metrics like the F1 statistic that try to capture more of what is going on, such that F1 = (2*True Positives [TP])/(2*TP + False Negatives + False Positives).

For segmentation, the author often uses the Dice similarity coefficient, which is 2 * (X ∩ Y)/(X + Y). Essentially, it is the overlap of the correct segmentation and the predicted segmentation. Another useful metric is the Hausdorff distance, which is the perpendicular distance between the edge of the correct segmentation and the predicted segmentation.

There are many more performance metrics used in AI, depending on the nature of the task and the data. It is important to select the correct metric when evaluating the performance of an AI tool.

Saliency maps and their relatives provide an indication of which parts of an image were important in making a prediction (**Fig. 3**). This serves at least 2 important purposes. First, it is a sanity check that the AI is making a decision based on a medically relevant part of an image. There now are several examples where saliency maps have shown AI tools making predictions using irrelevant portions of an image, such as the nature of a patient identification marker reflected a facility with a higher incidence of the disease being predicted. In other cases, the AI tool found that detecting chest tubes was a good way to detect

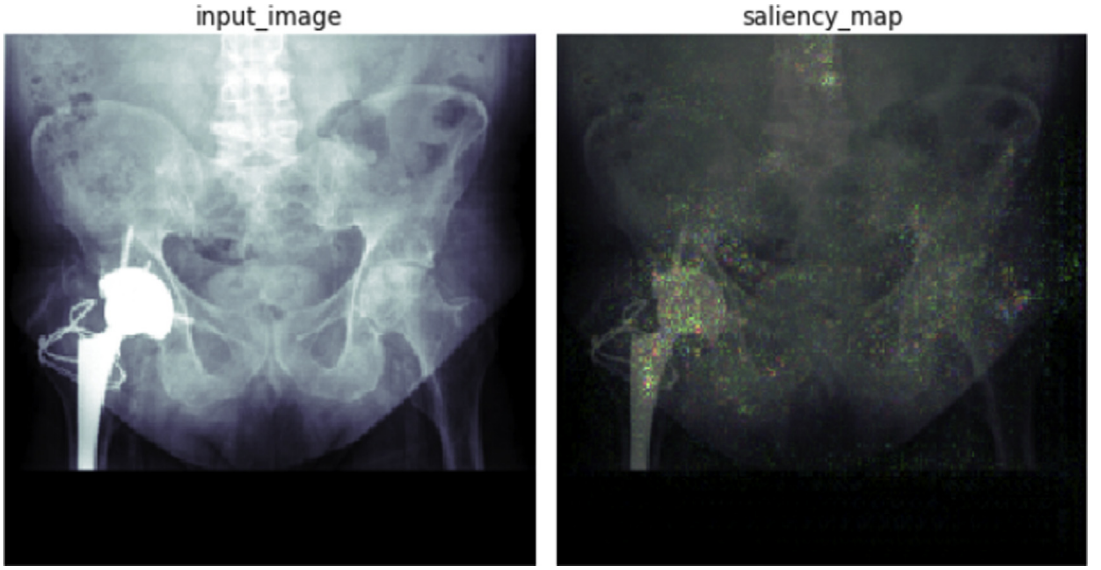

Fig. 3. An example saliency map (*right*). The deep learning algorithm was trained to find the total hip arthroplasty (*left*). Many activations overlying the implant that make sense can be seen. The other activations are not necessarily wrong and may reflect localization of the implant relative to the pelvis. There are some low-intensity activations outside the pelvis, and it is common to have such spurious hits as well. Most, however, should be in locations that make sense.

pneumothorax, but that is not clinically useful because the presence of a chest tube indicates the pneumothorax that already had been detected.

Adaptations of Neural Networks for Deep Learning in Medical Images

All the descriptions to this point are general and not specific to medical images. There are a few important distinctions between medical images and photographic images that should be considered when designing and training a network. First, there are advantages, such as the fact that the size of pixels and the orientation of the images usually are well known, and this can mean that fewer examples are required than with photographs of the real world. Second, in cases like CTs, the intensity scale is quite reproducible, whereas for most other imaging modalities, the intensity scale is not known, and some type of intensity normalization must be applied. There also can be shading in the image, where the intensity varies as a function of location, such as in MR imaging and field heterogeneity. Knowledge of the imaging modality, the anatomy, and the problem at hand is important in selecting the best approach to handling intensity.

Medical images also can have lots of other important associated information, and this still is a rather unexplored area. The simplest is that there can be images of the same anatomy but acquired with different properties—for example T1-weighted and T2-weighted images or images without and with intravenous contrast. There also may be old imaging examinations and the task may be to detect changes over time. Nonpixel information, such as age, gender, blood test results, and other information, can improve AI performance significantly. A description of how to apply these is beyond scope of this introduction but it is important to keep these factors in mind when planning an AI tool.

SUMMARY

AI technology has seen rapid advances in the past decade as deep technologies have enabled vastly superior performance to prior methods. Highly accurate image classification methods have shown superhuman level performance for some tasks. AI methods also can perform tedious tasks like outlining organs or tumors with human-level accuracy efficiently. It is likely that these will see adoption into clinical practice over the next decade as the strengths and weaknesses are better understood.

A common question is whether or not deep learning is always better than traditional machine learning. It is not. Deep learning demands large data sets because the many parameters it must learn easily can result in overfitting if too few examples are provided. Traditional methods should be selected when there are few training data, although there is not an exact number for what constitutes few. Some traditional machine learning methods like decision trees also have clear and easy-to-understand models, which can be important for convincing that the decisions made make sense and also may provide valuable insight into disease.

DISCLOSURE

B.J. Erickson is a founder and stockholder in FLowSIGMA, Inc.

REFERENCES

1. PS3.1. Available at: http://dicom.nema.org/medical/dicom/current/output/html/part01.html. Accessed January 15, 2021.
2. Milletari F, Navab N, Ahmadi S-A. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. arXiv [cs.CV]. 2016. Available at: http://arxiv.org/abs/1606.04797. Accessed July 12, 2018.
3. Robb RA, Hanson DP, Karwoski RA, et al. Analyze: a comprehensive, operator-interactive software package for multidimensional medical image display and analysis. Comput Med Imaging Graph 1989; 13:433–54.
4. Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature 1986;323:533–6.
5. Available at: https://papers.nips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf. Accessed January 15, 2021.
6. Krizhevsky A, Sutskever I, Hinton GE. ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira F, Burges C, Bottou L, Weinberger. KQ, editors.Advances in Neural Information Processing Systems 25. 2012. Available at: https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf. Accessed July 14, 2018.
7. Santurkar S, Tsipras D, Ilyas A, et al. How Does Batch Normalization Help Optimization? arXiv [stat.ML]. 2018. Available at: http://arxiv.org/abs/1805.11604. Accessed September 3, 2019.
8. Srivastava N, Hinton GR, Krizhevsky A, et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. J Machine Learn Res 2014;15: 1929–58.

9. Cogswell M, Ahmed F, Girshick R, et al. Reducing Overfitting in Deep Networks by Decorrelating Representations. arXiv [cs.LG]. 2015. Available at: http://arxiv.org/abs/1511.06068. Accessed November 12, 2018.
10. He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition. In: Bajcsy, editor. Proceedings of the IEEE Conference on computer Visions and Pattern Recognition. Los Alamitos (CA): Conference Publishing Services; 2016. p. 770–8.
11. Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv [cs.CV]. 2016. Available at: http://arxiv.org/abs/1602.07261. Accessed January 18, 2017.
12. Ronneberger O, Fischer P, Brox T. U-net: convolutional networks for Biomedical image segmentation. Medical image computing and Computer-Assisted Intervention – MICCAI 2015. Cham: Springer; 2015. p. 234–41.